4. Mitigation Firewalls

How many rules have been added to the firewall

- Check the auth.log file for entries related to adding firewall rules.

```
┌──(imen㉿hbtn-lab)-
[…/web_application_security/0x0c_web_application_foresics]
└─$ ./4-firewall.sh
6
```

Command Breakdown:

```
grep -iE "iptables" auth.log | grep "A INPUT" | sort -u | wc -l
```

1. **`grep -iE "iptables" auth.log`**:
   - This part searches the `auth.log` file for lines containing the word **`iptables`**. The **`-i`** flag makes the search case-insensitive, meaning it will match "iptables", "IPTABLES", "Iptables", etc. The **`-E`** flag allows extended regular expressions, although in this case it's not strictly necessary unless you're planning to add more complex patterns later.
   - This filters the log entries to only those involving `iptables`, which is a command-line utility used for configuring firewall rules in Linux.
2. **`grep "A INPUT"`**:
   - This filters the output of the previous `grep` command to only include lines that contain **`A INPUT`**. This pattern is used to match log entries related to **adding (`A`)** rules to the **`INPUT`** chain in `iptables`. The `INPUT` chain controls incoming network traffic.
   - This helps to focus on events where `iptables` rules are added to filter incoming traffic.
3. **`sort -u`**:
   - This sorts the filtered log entries in **ascending order** and removes any **duplicate entries**. The `-u` option ensures that only unique lines remain, so you won't count the same rule addition multiple times.
4. **`wc -l`**:
   - Finally, this counts the number of lines in the input, which corresponds to the number of **unique `iptables` rule additions** to the `INPUT` chain. Essentially, this gives the count of distinct log entries related to adding rules to control incoming traffic.

What the command does:

This entire command sequence counts how many unique rules have been **added** to the `INPUT` chain in `iptables` based on the entries in the `auth.log` file.

Example scenario:

Suppose your `auth.log` contains entries like:

```
Feb 24 14:23:56 server sshd[2345]: iptables: A INPUT -p tcp --dport 22 -j
ACCEPT
Feb 24 14:23:59 server sshd[2345]: iptables: A INPUT -p tcp --dport 80 -j
ACCEPT
Feb 24 14:24:05 server sshd[2345]: iptables: A INPUT -p tcp --dport 22 -j
ACCEPT
Feb 24 14:24:10 server sshd[2345]: iptables: A INPUT -p tcp --dport 443 -j
ACCEPT
```

After running the command:

1. It will search for entries with `iptables`, resulting in:

   ```
   iptables: A INPUT -p tcp --dport 22 -j ACCEPT
   iptables: A INPUT -p tcp --dport 80 -j ACCEPT
   iptables: A INPUT -p tcp --dport 22 -j ACCEPT
   iptables: A INPUT -p tcp --dport 443 -j ACCEPT
   ```

2. It will then filter for `A INPUT`, which is already done.
3. It will remove duplicate entries, leaving:

   ```
   iptables: A INPUT -p tcp --dport 22 -j ACCEPT
   iptables: A INPUT -p tcp --dport 80 -j ACCEPT
   iptables: A INPUT -p tcp --dport 443 -j ACCEPT
   ```

4. The final count (`wc -l`) will be:

   ```
   3
   ```

Use case:

This command is useful for tracking the number of unique times `iptables` rules have been added to the `INPUT` chain, specifically to control incoming traffic. It can help you monitor the configuration changes to the firewall, detect misconfigurations, or analyze the security policies applied to incoming network traffic.